Towards a software architecture for neurophysiological experiments

Ioannou, Constantina; Kindler, Ekkart; Bækgaard, Per; Saqid, Shazia; Weber, Barbara

Published in:
Proceedings of the NeuroIS Retreat 2019

Link to article, DOI:
10.1007/978-3-030-28144-1_17

Publication date:
2019

Document Version
Peer reviewed version

Towards a software architecture for neurophysiological experiments

Constantina Ioannou[1], Ekkart Kindler[1] Per Bækgaard[1] Shazia Saqid[2] and Barbara Weber[1,3]

[1] Technical University of Denmark, 2800 Kgs.Lyngby, Denmark
coio@dtu.dk, ekki@dtu.dk, pgba@dtu.dk
[2] The University of Queensland, St Lucia QLD 4072, Australia
shazia@itee.uq.edu.au
[3] University of St. Gallen, Switzerland
barbara,weber@unisg.ch

Abstract. Despite their wide adoption for conducting experiments in numerous domains, neurophysiological measurements often are time consuming and challenging to interpret because of the inherent complexity of deriving measures from raw signal data and mapping measures to theoretical constructs. While significant efforts have been undertaken to support neurophysiological experiments, the existing software solutions are non-trivial to use because often these solutions are domain specific or their analysis processes are opaque to the researcher. This paper proposes an architecture for a software platform that supports experiments with multi-modal neurophysiological tools through extensible, transparent and repeatable data analysis and enables the comparison between data analysis processes to develop more robust measures. The identified requirements and the proposed architecture are intended to form a basis of a software platform capable of conducting experiments using neurophysiological tools applicable to various domains.

Keywords: neurophysiological tools, software architecture, neurophysiological experiments

1 Introduction

Neurophysiological tools are used by researchers in numerous domains such as information systems (IS) and software engineering (SE) to measure human responses when people engage with information technology (IT) artifacts to perform a task (e.g. code comprehension, web browsing). The advantages of using these tools to measure human responses in the context of studies are the following: they provide objective measures, complementing subjective, perception-based measures and they enable continuous real-time data collection [6].

In addition, the availability of neurophysiological tools at a better quality, lower cost and reduced intrusiveness motivated researchers to apply such measurements (e.g., to

assess cognitive load). Cognitive load provides insights into a person's processing capabilities while performing a task which can support personalized IS design and adaptation to the users' cognitive needs (e.g., in a digital learning context the task difficulty could be decreased when high cognitive load is observed to avoid stress, frustration, and errors) [6]. Therefore, research has focused on creating reliable and objective measures of cognitive load [4]. For instance, a research focused on pupillary response data for identifying an indicator of cognitive load [7], while another study used multi-modal measurements (e.g., the combination pupillary responses with EDA signals) to derive cognitive load and to assess task difficulty [8].

In response to these trends, software solutions have evolved to support researchers during the execution of experiments including the collection and synchronisation of neurophysiological measurements as well as the (online) analysis of data. Software solutions like Brownie, iMotions, CubeHX, Noldus Observer, and OpenVibe support the design of experiments and the collection and synchronisation of neurophysiological measurements including their visualization [9,10,12,13,14]. Moreover, software solutions like EEGLAB and OpenVibe provide extensible and transparent ways to analyse data and partially support repeatable data analysis pipelines [5,14].

However, current solutions do not provide any automatic means to compare different data analysis pipelines to obtain robust measures (e.g., of cognitive load). In order to close this gap, we aim to identify a set of requirements and develop a software architecture that not only supports comparing different analysis pipelines, but additionally also supports multi-modal neurophysiological measurements in an extensible, transparent, and repeatable way.

2 Requirements

In recent years, an increasing body of literature applies neurophysiological measurements to investigate cognitive and emotional states of a developer during software engineering tasks [7,8,11]. By using this domain as a basis, we derived a group of selected requirements for a software architecture that aims to provide support for researchers in defining, deploying, executing, and analysing experiments using neurophysiological measurements. In this section, first we explain the requirements and briefly discuss related work.

Multi-modal measurements collection. Various studies aimed at better understanding the cognitive load of developers while engaging with different software artifacts using neurophysiological measurements. For example, one study used pupillary response data, electroencephalography (EEG) and galvanic skin response (GSR) to assess task difficulty during change tasks [8]. Similarly, another study combined EEG and pupillary response data to predict task difficulty during program comprehension tasks [11]. In all these studies multiple neurophysiological measurements were collected and had to be associated with the task context. Therefore, we formulate our first requirement as follows:

> **R1:** A software architecture for neurophysiological experiments needs the ability to support the simultaneous multi-modal measurement collection, including their synchronisation and their association with the experiment's task context.

Considering the need for synchronising different neurophysiological measurements, several solutions have evolved (e.g.,Brownie, iMotions, Noldus Observer, CubeHX, and OpenVibe) [9,10,12,13,14]. These solutions ease the complexity and the time required to collect multi-modal data and associate it with the task context.

Extensible data analysis. Neurophysiological tools enable to obtain objective, continuous real-time measurements as a basis to determine the cognitive load of a subject. However, analysing such data is often challenging because it is notoriously noisy, requires cleaning before analysis and is difficult to interpret (e.g., discriminate the specific cause of the cognitive load changes) [3]. Moreover, there is no agreed upon way how to best measure cognitive load. To deal with these difficulties, a plethora of research aims to better understand cognitive load changes by analysing individual and combined modalities while using software artifacts [2,8]. More, recently data-driven approaches based on machine learning became increasingly popular to link neurophysiological data and measures of interest [1]. Thus, extensible support for the processing of data is needed. Our second requirement is formulated as:

> **R2:** A software architecture for neurophysiological experiments needs to be extensible in two ways: first, by supporting the ability to adopt external customized cleaning and analysis processes and second, by supporting the ability to include new devices and modalities.

Some software solutions (e.g., EEGLAB and OpenVibe) have emerged, that provide support for data analysis and enable researchers to incorporate their own cleaning and analysis processes [5,14].

Transparent data analysis. Interlinked with the previous requirement of extensible data analysis, is the need for transparent data analysis. Transparent data analysis means to be able to trace how a particular measure was obtained from the raw neurophysiological measurements collected by the used neurophysiological tools. Thus, our third Requirement is formulated as follows:

> **R3:** A software architecture for neurophysiological experiments needs to be transparent. The functionality of analysers and the flow of analysis processes should be visible to enable better understanding and optimization of these processes.

Current software solutions which support data analysis processes are domain and measurement specific. For example, EEGLAB and OpenVibe provide visibility and enable modifications of their analysis processes and their combinations [5,14]. However, they primarily focus on EEG. On the contrary, other solutions (e.g., iMotions and CubeHX [10,12]) offer more generic solution providing support for conducting multi-

modal neurophysiological experiments. However, they are proprietary software solutions and it often remains opaque to the researcher how measures are derived from the raw neurophysiological measurements.

Repeatable data analysis. Deriving robust measures from raw neurophysiological measurements is often challenging because of the mapping of neurophysiological measures to theoretical constructs [6]. For instance, cognitive load can be measured using pupillary response data, but also using EEG or GSR data or combinations thereof [7,8]. However, it is still unclear which (combination) of modalities and measures derived from them is the most robust and suitable for a particular setting. The application of data-driven methods based on machine learning to develop measures for theoretical constructs is getting increasingly popular [1,8]. Therefore, we formulate our fourth requirement as follows:

> **R4:** A software architecture for neurophysiological experiments needs to support repeatable data analysis, which allows data from previously collected neurophysiological experiments to be replayed and analysed with different analysis processes as well as applying pre-defined analysis processes on different data sets. Moreover, it needs to support the systematic testing and comparison of different analysis processes to develop more robust measures, even in near-realtime settings.

Software solutions like EEGLAB and OpenVibe support repeatable experiments by enabling the replay of data sets and the application of different data analysis processes including documentation of the process followed [5,14]. However, the comparison between measures is non-automatic and, therefore determining a robust and suitable measure is non-trivial.

3 Proposed Software Architecture

We followed the design science research approach to develop the architecture [15]. To do that, we decomposed the architecture into subproblems based on the requirement list. To satisfy the list of requirements, we conducted a few engineering cycles and an architecture which is highly configurable and extensible, in the sense that a variety of components can be combined or included, has emerged.

Fig. 1 shows the proposed architecture including its components which support researchers to define, deploy, execute and analyse experiments. The architecture consists of five main components: the Experiment Planning Component (EPC), the Data Acquisition Component (DAC), the Analysis Component (AC), the Storage Component (SC) and the Dispatcher Component (DC).

The EPC provides a user interface allowing researchers to define and deploy experiments including neurophysiological measurements. Moreover, the EPC allows researchers to specify reusable analysis pipelines. The DAC facilitates the collection of multi-modal measurements and the association of those with the task context, in a synchronised manner. The DAC consists of two providers: the sensing and the context provider. The sensing providers can cover user interactions (i.e. mouse clicks) and

physiological measures (i.e. eye movements, skin conductance). The context providers, in turn, register events during the execution of an experimental task, e.g., events marking the start and end of sub-tasks (i.e. reading code).

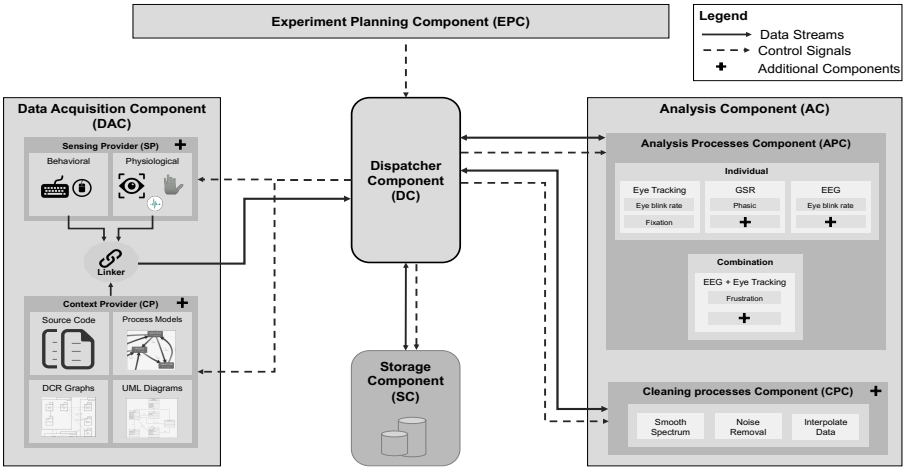

Fig. 1. The proposed architecture based on the selected requirements

The AC contains a vast amount of cleaning and analysis processes while new processes which extract new or existing measures can be incorporated. The AC enables modular analysis pipelines meaning that analysis pipelines can be easily set up, e.g., a pipeline to extract the eye blink rate from a stream of raw eye tracking data or to characterize pupillary responses to presented stimuli, and to replace cleaning and analysis processes in the pipeline to compare the behaviour of different processes.

The SC is responsible to store the collected raw data, the task context, the meta-data of the experiment (i.e, sampling rates and devices), the results of data analysis, and the applied analysis pipelines.

Finally, the DC is a scheduling component and is responsible to ensure that the experiment is executed according to its definition. To do that, it controls the data flow between the components (i.e., to retrieve, forward, and store data). Furthermore, the DC can instantiate multiple data analysis pipelines to run in a parallel manner.

Next, we will present the interplay of the different components using the following scenarios which are simple examples of how the software architecture can be useful.

- Scenario 1: As a researcher, I would like to conduct (define, deploy, execute, and analyse) experiments using neurophysiological measurements, e.g., combining pupillary response data obtained from eye tracking and EDA signals obtained from GSR.
- Scenario 2: As a researcher, I would like to replay previously collected data from an experiment and I would like to use different analysis processes to determine, for example, which one is able to predict the subject's task performance best.

Fig. 2 presents the interplay of components for Scenario 1. The experiment uses eye tracking and GSR measurements to investigate the understandability of IT artifacts (collected by DAC). The collected raw measurements are received by the DC and stored. Next, these data are forwarded to the AC components where cleaning and analysis processes take place. After each processing step the DC stores the results and potentially visualizes them.

In Fig. 3, the interplay of components for Scenario 2 is shown. The configured pipeline represents an experiment where previously collected data (from Scenario 1) will be replayed to investigate the understandability of software artifacts, but in this case different analysis pipelines are used. For example, different analysis processes could be used to test which combination is better able to predict the subject's task performance.

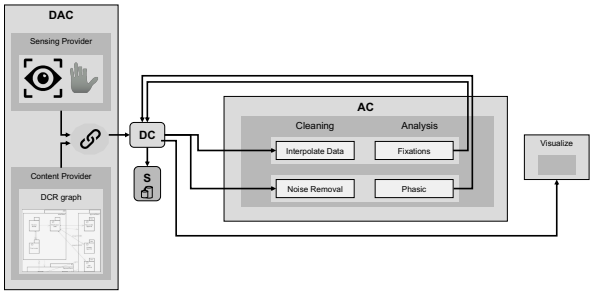

Fig. 2. Interplay of components: Scenario 1

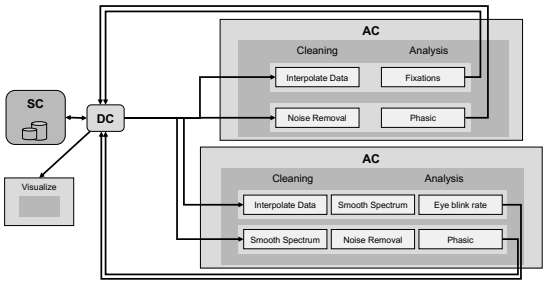

Fig. 3. Interplay of components: Scenario 2

The envisioned architecture supports researchers during all stages of the experimental cycle, i.e., experiment planning, experiment execution, and experiment analysis by fulfilling the requirements set in Section 2 [16]. The suggested architecture supports *Multi-modal measurements collection (R1)* by enabling the collection of neurophysiological measurements in a synchronised manner linked with the task context (cf. DAC). Moreover, it provides a modular architecture supporting *Extensible data analysis(R2)* with sub-components each representing a different data analyser (cf. AC). Additional components can be added without influencing the functionality of other components.

The analysis components are transparent (*Transparent data analysis(R3))* due to their open-source nature. Moreover, data analysis pipelines can be built that define the data analysis process in a reusable manner using the EDC and can be later traced since a history of those is maintained by the SC. Finally, the architecture supports *Repeatable data analysis(R4)* since data can be replayed (cf. DC) and experiments can be repeated using different analysis pipelines. Moreover, existing analysis pipelines can be reused for different data sets. Additionally, the DC ensures the capability of comparing analysis pipelines, because it is designed for instantiating several analysis processes and run those in a parallel manner. In conclusion, the proposed architecture supports the collection of neurophysiological measurements in a synchronised manner and supports the extensible, transparent, and repeatable analysis of data including the comparison of analysis processes.

4 Conclusions

This paper elaborates on the requirements and describes a preliminary software architecture that aids researchers conducting (defining, deploying, executing and analysing) neurophysiological experiments in an extensible, transparent and repeatable manner. While the collection and synchronisation of neurophysiological measurements is already well supported by existing software solutions, most software solutions only provide limited support to the extensible, transparent, and repeatable analysis of neurophysiological data. In particular, current solutions do not provide automatic means to systematically compare different data analysis pipelines to obtain robust measures. As a next step, we will start implementing a software framework for conducing neurophysiological experiments based on the proposed architecture (reusing available components where possible). The software framework, while emerging from our own research in the SE and NeuroIS fields, is not specific to these communities and has the potential for contributing in other research fields that have the challenge of obtaining robust measures from neurophysiological measurements.

References

1. Bednarik, R., Vrzakova, H., and Hradis, M.: What Do You Want to Do Next: A Novel Approach for Intent Prediction in Gaze-based Interaction. In: Proceedings of the Symposium on Eye Tracking Research and Applications. Pp. 83-90. ETRA '12, ACM, New York, NY, USA (2012). http://doi.acm.org/10.1145/2168556.2168569, http://10.1145/2168556.2168569
2. Buettner, R., Sauer, S., Maier, C., Eckhardt, A.: Real-time prediction of user performance based on pupillary assessment via eye-tracking. Ais Transactions on Human-computer Interaction pp. 26-60 (2018). http://10.17705/1thci.00103, 10.17705/1thci
3. Chen, F., Ruiz, N., Choi, E., Epps, J., Khawaja, M., Taib, R., Yin, B., Wang, Y.: Multimodal behavior and interaction as indicators of cognitive load. ACM Transactions on Interactive Intelligent Systems (TiiS) 2 (12 2012). https://doi.org/10.1145/2395123.2395127
4. Chen, F., Zhou, J., Wang, Y., Yu, K., Arshad, S., Khawaji, A., Conway, D.: Robust multimodal cognitive load mearurement. In: Human–Computer Interaction Series (2016)

5. Delorme, A., Makeig, S.: Eeglab: An open source toolbox for analysis of single-trial eeg dynamics including independent component analysis (2011). https://doi.org/10.1.1.196.2963
6. Dimoka, A., Banker, R.D., Benbasat, I., Davis, F.D., Dennis, A.R., Gefen, D., Gupta, A., Ischebeck, A., Kenning, P.H., Pavlou, P.A., Maller-Putz, G., Riedl, R., Vom Brocke, J., Weber, B.: On the use of neurophysiological tools in is research: Developing a research agenda for neurois. Mis Quarterly: Management Information Systems 36 (3), 679–702 (2012)
7. Duchowski, A.T., Krejtz, K., Krejtz, I., Biele, C., Niedzielska, A., Kiefer, P., Raubal, M., Giannopoulos, I.: The index of pupillary activity: Measuring cognitive load vis-`a-vis task difficulty with pupil oscillation. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. pp. 282:1-282:13. CHI '18, ACM, New York, NY, USA (2018) http://doi.acm.org/10.1145/3173574.3173856
8. Fritz, T., Begel, A., Müller, S.C., Yigit-Elliott, S., Züger, M.: Using psycho-physiological measures to assess task difficulty in software development. In: Proceedings of the 36th International Conference on Software Engineering. pp. 402–413. ICSE 2014, ACM, New York, NY, USA (2014) http://doi.acm.org/10.1145/2568225.2568266
9. Hariharan, A., Lux, E., Pfeiffer, J., Dorner, V., Maller, M.B., Weinhardt, C., Adam, M.T.P.: Brownie: A platform for conducting neurois experiments. Journal of the Association of Information Systems 18 (4), 264–296 (2017)
10. iMotions: https://imotions.com/ (2018)
11. Lee, S., Hooshyar, D., Ji, H., Nam, K., Lim, H.: Mining biometric data to predict programmer expertise and task difficulty. Cluster Computing pp. 1–11 (1 2017).
12. Léger, P.M., Courtemanche, F., Fredette, M., Sénécal, S.: "A cloud-based lab management and analytics software for triangulated human-centered research". Lecture Notes in Information Systems and Organisation 29, pp 93–99 (2019).
13. NOLDUS, L.: The observer - a software system for collection and analysis of observational data. Behavior Research Methods Instruments and Computers 23 (3), 415–429 (1991)
14. Renard, Y., Lotte, F., Gibert, G., Congedo, M., Maby, E., Delannoy, V., Bertrand, O., Lécuyer, A.: Openvibe: An open-source software platform to design, test, and use braincomputer interfaces in real and virtual environments. PRESENCE: Teleoperators and Virtual Environments 19, pp 35–53 (2010)
15. Wieringa, R.: Design science methodology for information systems and software engineering. Springer (2014). https://doi.org/10.1007/978-3-662-43839-8
16. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslé, A.: Experimentation in software engineering. Experimentation in Software Engineering, 1–236 (2012). https://doi.org/10.1007/978-3-642-29044-2